RESEARCH PAPER

Link Prediction based on Tensor Decomposition for the Knowledge Graph of COVID-19 Antiviral Drug

Ting Jia[1]*, Yuxia Yang[1]*, Xi Lu[1], Qiang Zhu[1], Kuo Yang[1,2†] & Xuezhong Zhou[1,†]

[1]Institute of Medical Intelligence, School of Computer and Information Technology, Beijing Jiaotong University, Beijing 100044, China

[2]BNRIST/Department of Automation, Tsinghua University, Beijing 100084, China

Keywords: Link prediction; Knowledge graph; COVID-19; Antiviral drug prediction; Tensor decomposition

Citation: Jia, T., et al.: Link prediction based on tensor decomposition for the knowledge graph of COVID-19 antiviral drug. Data Intelligence 4(1), 134-148 (2022). doi: 10.1162/dint_a_00117

Received: February 14, 2021; Revised: June 10, 2021; Accepted: August 17, 2021

ABSTRACT

Due to the large-scale spread of COVID-19, which has a significant impact on human health and social economy, developing effective antiviral drugs for COVID-19 is vital to saving human lives. Various biomedical associations, e.g., drug-virus and viral protein-host protein interactions, can be used for building biomedical knowledge graphs. Based on these sources, large-scale knowledge reasoning algorithms can be used to predict new links between antiviral drugs and viruses. To utilize the various heterogeneous biomedical associations, we proposed a fusion strategy to integrate the results of two tensor decomposition-based models (i.e., CP-N3 and ComplEx-N3). Sufficient experiments indicated that our method obtained high performance (MRR=0.2328). Compared with CP-N3, the mean reciprocal rank (MRR) is increased by 3.3% and compared with ComplEx-N3, the MRR is increased by 3.5%. Meanwhile, we explored the relationship between the performance and relationship types, which indicated that there is a negative correlation (PCC=0.446, *P*-value=2.26e-194) between the performance of triples predicted by our method and edge betweenness.

1. INTRODUCTION

Knowledge graph (KG) is a structured representation of real-world information. In KGs, nodes represent entities, such as people and places; edges are specific facts that connect two entities; labels are the types of edges [1, 2]. As a basis for knowledge engineering applications, KG plays an extremely important role

* Ting Jia and Yuxia Yang contributed equally to the work.

† Corresponding authors: Xuezhong Zhou (Email: xzzhou@bjtu.edu.cn; ORCID: 0000-0002-4713-3594) and Kuo Yang (Email: yangkuo@bjtu.edu.cn; ORCID: 0000-0003-0736-4512).

for various artificial intelligence tasks (e.g., clinical decision making and question-answering systems). However, it is well known that even the most advanced KGs still suffer from incompleteness [1,3, 4], such as FreeBase [5], WikiData [6], DBPedia [7] and Google KG [8]. Link prediction (LP), which exploits the existing facts in a KG to infer missing ones, is one of the most prospective ways to address this problem [8]. At the time of the large-scale spread of COVID-19 [9], China Conference on Knowledge Graph and Semantic Computing (CCKS) 2020 set up a link prediction task for the knowledge graph of COVID-19 antiviral drug (ADKG), requiring prediction of the associations between two biomedical entities in the graph. It has great significance and academic value for the research and development of antiviral drugs.

At present, in terms of link prediction tasks, methods that can be considered include the similarity of node attribute, network structure, likelihood analysis and machine learning. Of these methods, some are based on classification methods, such as decision trees [10], support vector machines [11], and multi-layer perceptions [12]. and some are based on probability graph models and matrix decomposition methods [13, 14, 15, 16]. In recent years, tensor decomposition has attracted much attention [14]. The Canonical Tensor Decomposition (CP) decomposes a high-order tensor into several one-dimensional factor matrices [13, 14, 15, 16, 17]. For example, a 3rd-order tensor can be decomposed into three one-dimensional factors. It can be understood as a low-rank approximation or feature extraction of high-dimensional data. ComplEx [16], a complex number-based representation method can grasp the symmetric and asymmetric relationships in the binary relationship of the KG. However, how to design an effective strategy of fusing different link prediction algorithms is still a challenge. In addition, it is not clear how different relationship types and even different test triples show different performances in link prediction algorithms.

Therefore, we introduced ensemble learning into the link prediction task. By using the existing methods, we adopted ensemble ideas and combined multiple supervision models to obtain a better and more comprehensive supervised model. We participated in the CCKS 2020 link prediction task and proposed a fusion strategy, which effectively integrated the results obtained by two tensor-decomposition models (i.e., CP-N3 and ComplEx-N3). The experimental results indicated that our method obtained high performance (MRR=0.2328), which ranked third in the fierce competition of CCKS 2020 Challenge Task 1. At the same time, in order to explore whether the prediction performance is related to the relationship type in the real world data, we analyzed the prediction results of different relationship types in the benchmark data. From the perspective of complex networks, we explored the relationship between the prediction results of different entities and the edge measurement, i.e., edge betweenness.

2. RELATED WORK

In general, there are two approaches to the task of link prediction on KG [18]: embedding-based approach [16, 19, 20] and rule-based approach [18, 21, 22]. On the embedding-based methods, the entities and relations in the KG are mapped to low-dimensional space. Rule-based methods attempt to capture the inherent regularities in a KG as rules [18], and then use the learned rules to reason on the KG.

TransE [19] is a classical translational distance model, which regards the relation as a translation from the head entity to the tail entity. Inspired by TransE, many variants have been proposed, e.g., STransE [23] and RotatE [20]. With the development of deep learning, neural network-based models have gradually been applied to LP task [24], which are conducted by using neural networks to calculate the scoring functions of triples. ConvE [24] is a multi-layer convolutional network which aims to extract higher-level, non-linear features. R-GCN [25] is developed specifically to handle the highly multi-relational data characteristics of KG. RSN [26] combines RNN with residual learning to learn relation paths, which effectively captures the long-term relational dependencies. In recent studies, tensor decomposition-based methods have attracted much attention [13, 14]. Representation learning based on tensor decomposition treats each triple as an element in the tensor and performs representation learning through a tensor decomposition algorithm. In these methods, tensor is a combination of low-dimensional vectors, which are used as embeddings of entities and relationships. As the first typical representative algorithm that uses tensor decomposition for the completion of knowledge graphs, RESCAL [27] uses a 3rd-order tensor to represent entities and relationships. However, due to the large number of parameters, the risk of overfitting is more likely to occur, so the DistMult [15] algorithm appears. This method proposed that each type of relationship can be represented as a diagonal matrix. While reducing the space of parameters to be learned, DistMult [15] introduces a new problem, that is, it treats all relations as symmetric. ComplEX [16] extends DistMult [16] in the complex space, which uses vector dot multiplication to calculate the triplet score and uses probability to determine the correctness of the triplet. It can also model asymmetric relationships. CP [13] regards the link prediction as a 3rd-order binary tensor completion problem, which embeds the head and tail entities of the unified entity independently to each other.

In addition to these embedding-based methods, rule-based methods have also received much attention. Some path-based methods take a set of paths as input which are gathered by performing random walks. MINERVA [22], eliminating any need for precomputed paths, trains an agent by reinforcement learning to walk to the answer node conditioned on the input query. RNNLogic [21] learns logic rules for reasoning on KGs. It treats logic rules as a latent variable, and simultaneously trains a rule generator as well as a reasoning predictor with logic rules. EARDict [18] provides a fundamental theory for KG reasoning based on the *ending anchored rules*, which provides a large amount of rules to evaluate the correctness of unknown triples.

3. METHODS

3.1 Data Preprocessing

First, we used regular expressions to match the officially given data files to get the data with triple format we need. Second, we added the list of entity attributes to our knowledge graph to get more attribute information of the entity. Finally, to expand more relationships from the knowledge graph, we used a well-known data augmentation technique [13, 24], which added reciprocal relations for every triple, i.e., adding (t, r^{-1}, h) for every (h, r, t), where h is the head entity, r is the relationship and t is the tail entity.

3.2 CP-N3

CP [17] represents a tensor $\mathcal{X} \in R^{N_1 \times N_2 \times N_3}$ as a sum of R rank one tensors $u_r^{(1)} \otimes u_r^{(2)} \otimes u_r^{(3)}$ ($\otimes$ denotes tensor product) where $r \in \{1, \ldots, R\}$ and $u_r^{(m)} \in R^{N_m}$:

$$\mathcal{X} = \sum_{r=1}^{R} u_r^{(1)} \otimes u_r^{(2)} \otimes u_r^{(3)} \tag{1}$$

The KG completion performance of current CP on the standard benchmarks are behind their competitors. Timothee et al. [13] proposed a novel regularizer, based on tensor nuclear p-norms, and then presented a reformulation of the problem, which makes it beat the current state-of-the-art on several datasets with a CP decomposition (named CP-N3).

3.3 ComplEx-N3

Similar to DistMult [15], ComplEx [16] represents each type of relation as a diagonal matrix, but extends the vector space into the complex space: $h \in C^d$, $t \in C^d$, and $r \in C^{d \times d}$. In the complex space, the bilinear product becomes a Hermitian product, where in lieu of the traditional t, its conjugate-transpose $\overline{t}$ is employed. This allows ComplEx to successfully model asymmetric relations as well. Timothee et al. [13] proposed a ComplEx-based model (named ComplEx-N3) with the novel regularizer and systematic parameter searches, which obtained better performance than ComplEx.

3.4 A Fusion Strategy

Different from the traditional machine learning method with one learner to train, the integrated learning method usually trains multiple learners and combines them to obtain better results. These methods have been applied to a variety of fields, such as computer vision [28], species distribution prediction [29] and auxiliary diagnosis [30].

Inspired by the idea of integrated learning, we designed a fusion strategy for this task (Figure 1). First, we use the same KG to train the CP-N3 and ComplEx-N3 models, respectively. Second, we use the two models to predict the results separately and get the top 20 entities with the highest score based on the predicted score. Third, we assign weights from 20 to 1 according to the ranking of the entities. For example, the first-ranked entity has a weight of 20, the second-ranked entity has a weight of 19 and so on. Finally, the two top 20 results are weighted integrated by voting to obtain the final top 10 predictive results. When the results are integrated, we restrict the entity based on the entity type, and eliminate the entities that are not of this type.

Figure 1 shows the framework of our fusion strategy. First, we use the same KG to train the CP-N3 and ComplEx-N3 models, respectively. Second, we use the two models to predict the results separately and get the top 20 entities with the highest score based on the predicted score. Third, we assign weights from 20 to 1 according to the ranking of the entities. For example, the first-ranked entity has a weight of 20, the second-ranked entity has a weight of 19 and so on. Finally, the two top 20 results are weighted integrated by voting to obtain the final top 10 predictive results.

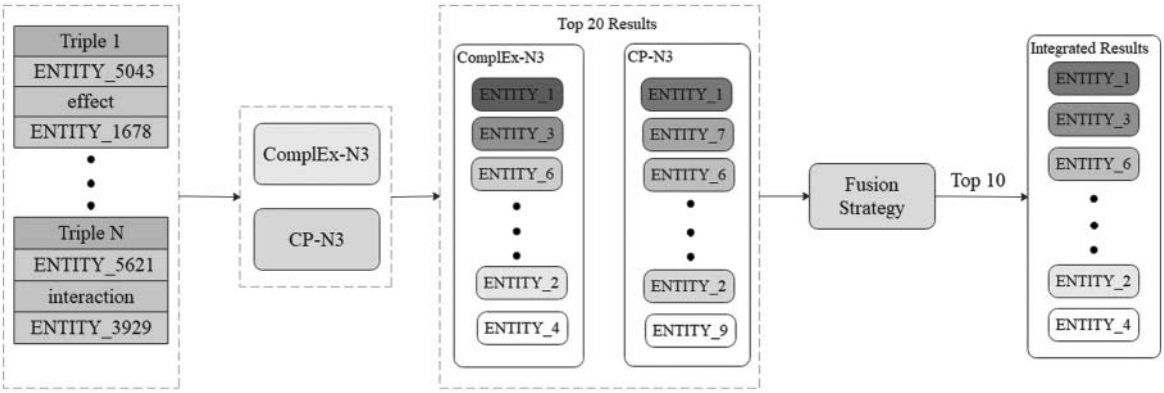

Figure 1. The framework of our fusion strategy.

4. EXPERIMENTS

4.1 Datasets

We used the ADKG of CCKS 2020 to evaluate our approach. The dataset was published by the CCKS 2020 Task 1, which includes three types of entities (i.e., proteins, viruses, and drugs) and four types of relationships (i.e., effect, interaction, produce and binding). ADKG includes 7,844 entities and 40,000 triples. In addition, there are two batches of prediction sets. The first batch of prediction set has 4,256 triples and the second batch of prediction set has 5,000 triples. We call them prediction sets because the triples in the datasets lack the corresponding head entity or tail entity, i.e., $(h, r, ?)$ or $(?, r, t)$. Our task is to predict the corresponding entities to the missing part based on the trained model. To evaluate the performance of models, we randomly selected 10% of each relationship type as the test set, resulting in 36,000 triples in the training set and 4,000 triples in the test set. In order to get more attribute information of the entities, we added the list of entity attributes to the training set and added the involved entities to the attribute entity list simultaneously. In total, the knowledge graph has 8,494 entities and 51,131 triples, of which 47,131 triples are in the training set and 4,000 triples are in the test set.

4.2 Experimental Setup

We tuned all hyper-parameters based on predictive performance on test set to find the best parameter settings on ADKG. Table 1 shows the best parameter settings of the two methods. In experiments, we used the well-tuned parameters to obtain the top 20 results of the two methods, respectively, and integrated the two results according to our fusion strategy to obtain the final results.

Table 1. Best parameter settings of CP-N3 and ComplEx-N3 on ADKG.

Hyper-parameters	CP-N3	ComplEx-N3
Dimension	2000	2500
Learning rate	0.06	0.07
Batch size	256	128
Regularization coefficient	0.01	0.005
Initialization	0.001	0.001
Optimizer	Adagrad	Adagrad

5. RESULTS

5.1 Comparison Results with Baselines

For each test triple, we generated 8,494 candidate triples by combining the test entity-relation pair with all possible entities E, ranking the scores obtained. We used the filtered setting [19], i.e., all known true triples are removed from the candidate set except for the corresponding entities in current test triple. Since the top 10 candidate entities were finally submitted, we still cared about the predicted entities after the top 10 to integrate the results. Therefore, we used mean reciprocal rank (MRR) of top 10 (MRR@10) and Hits@k, $k \in \{10, 20, 50, 100\}$ to evaluate the performance. MRR@10 is the average of the inverse of the obtained ranks over the top 10 candidate triples. Hits@k measures the percentage of times a true triple is ranked within the top k candidate triples. Comparison results of link prediction on test set of ADKG were shown in Table 2. In addition to the two given baselines (i.e., TransE [19] and R-GCN [25]), we also showed the performance of other six classic methods, i.e., DistMult [15], ComplEx [16], TuckER [14], HypER [31], CrossE [32] and ConvE [24]. Results showed that CP-N3 (MRR@10=0.185) and ComplEx-N3 (MRR@10= 0.183) not only achieved better than some linear models (e.g., DistMult, ComplEx and TuckER), but also better than some deep neural network (e.g., RGCN, HypER, CrossE and ConvE). The performance of CP-N3 and ComplEx-N3 far exceeds other methods, respectively. Therefore, we used these two methods as our base learners and integrated their prediction results based on our fusion strategy (MRR@10=0.218). In the

Table 2. Comparison results of link prediction methods.

Model	Hits@10	Hits@20	Hits@50	Hits@100	MRR@10
TransE [19]	0.17	0.24	0.34	0.42	0.076
R-GCN [25]	0.20	0.26	0.38	0.49	0.079
DisMult [15]	0.301	0.384	0.506	0.596	0.140
ComplEx [16]	0.308	0.391	0.504	0.588	0.145
TuckER [14]	0.337	0.428	0.545	0.640	0.166
HypER [31]	0.328	0.418	0.546	0.638	0.161
CrossE [32]	0.251	0.332	0.440	0.531	-
ConvE [24]	0.311	0.402	0.524	0.615	0.150
ComplEx-N3 [13]	0.375	0.462	0.589	0.674	0.183
CP-N3 [13]	0.375	0.467	0.589	0.676	0.185
Fusion Strategy (ours)	**0.414**	**0.503**	**0.619**	**0.691**	**0.218**

fierce competition of CCKS 2020 Challenge Task 1, it scored 19.785 (ranking first) in the first batch of predictive data and scored 23.275 (ranking third) in the second batch of data. In the competition, MRR@10*100 is used as the evaluation score.

5.2 Parameter Sensitivity

To investigate the influence of hyper-parameters of CP-N3 and ComplEx-N3, we tuned all related hyper-parameters of them to find the best parameter settings on ADKG. We selected Hits@k, $k \in$ {10, 20, 50, 100}, MRR and MRR@10 as evaluation metrics. There are four parameters that need to be tuned. The first parameter is the entity and relation dimension of the embedding vectors learned in the model, tuning from {500, 1000, 1500, 2000, 2500}. The second parameter is learning rate, which determines the decrease rate of parameters during optimization, tuning from {0.05, 0.06, 0.07, 0.08, 0.09, 1.0}. The third parameter is batch size, indicating the sample size for training at the same time, tuning from {128, 256, 512, 1024}. The final parameter is regularization coefficient, tuning from {0.005, 0.01, 0.05, 0.1}. In turn, we fixed each possible value of the three parameters, while tuning the remaining parameter.

Experimental results are shown in Figure 2 (A-D for CP-N3 and E-H for ComplEx-N3). In CP-N3, whether the batch size is too small (128) or too large (512 or 1024), it shows poor performance, but good performance

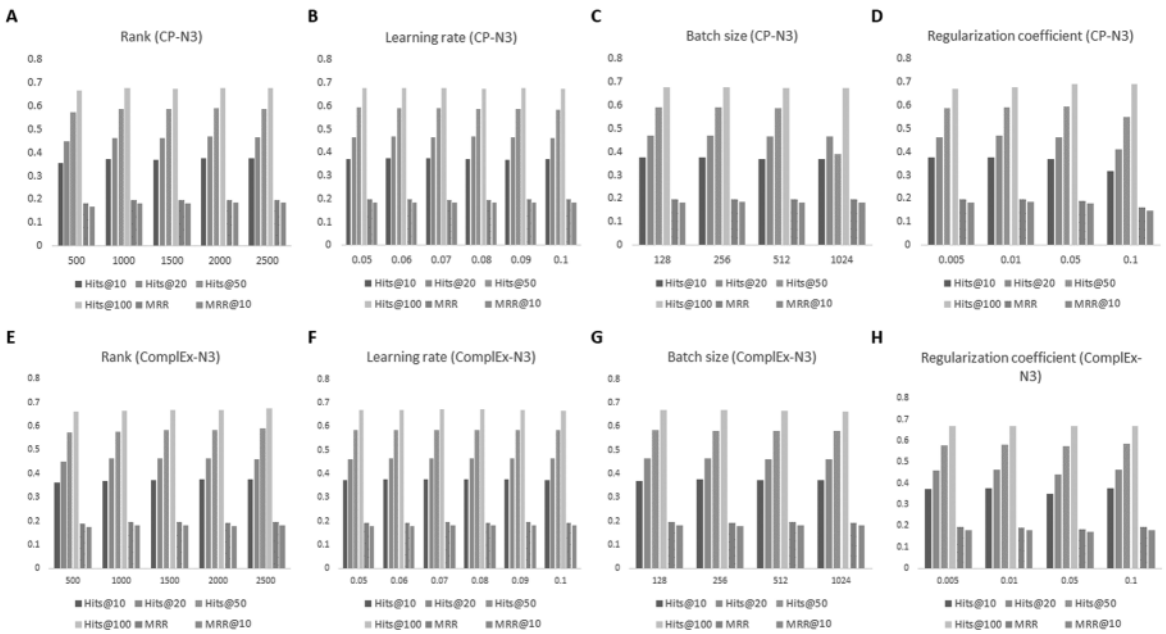

Figure 2. The prediction performance with tuning the hyper-parameters. The first parameter is the entity and relation dimension of the embedding vectors, tuning from {500, 1000, 1500, 2000, 2500}. The second parameter is learning rate, tuning from {0.05, 0.06, 0.07, 0.08, 0.09, 1.0}. The third parameter is batch size, tuning from {128, 256, 512, 1024}. The final parameter is regularization coefficient, tuning from {0.005, 0.01, 0.05, 0.1}. A-D sub-figures are for CP-N3 and E-H sub-figures are for ComplEx-N3.

is obtained when the level is equal to 256. The dimension between 1000 and 2500 does not have much influence on the model, and the small-scale adjustment of learning rate has little influence. In terms of prediction performance, it shows that the model has strong robustness in predicting candidate entities. Similarly, in ComplEx-N3, the best performance is achieved when the batch size is 128, the dimension is 2500, the regularization coefficient in {0.005, 0.01, 0.1} and the learning rate in {0.07, 0.09}.

In order to investigate the influence of the fusion number on the experimental performance, we conducted the experiments with different Top *n* selected (the range is {10, 20, 30, 40, 50, 60, 70, 80, 90, 100}). The results in Figure 3 show that with the increase of the fusion number, the prediction performance is improved, and the algorithm obtains the highest Hits@10 and MRR at top 30 and 20, respectively. At the same time, it also shows that after reaching the top 50, the prediction performance is reduced to different degrees. Therefore, combining the results of these two metrics, the fusion number we finally selected is top 20.

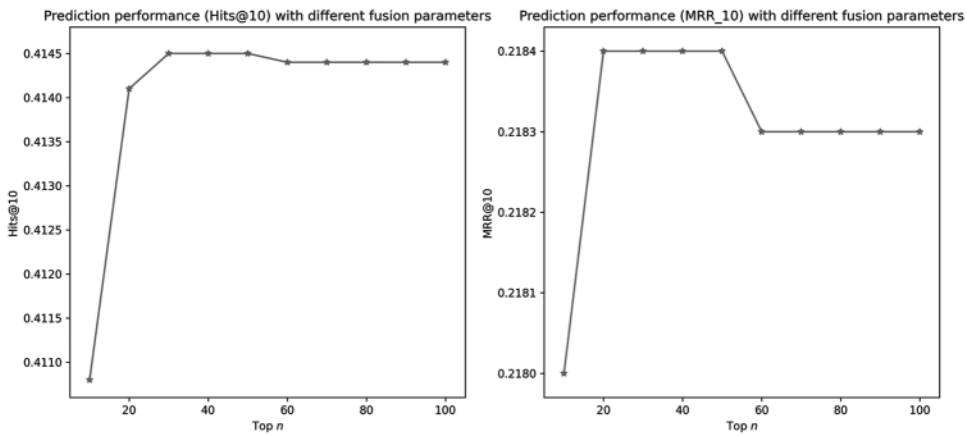

Figure 3. The prediction performance with fusing different top *n* values. The left sub-figure is the result of hits@10. It has a turning point at Top 20, reaches the optimal value at Top 30, and then stabilizes. The right sub-figure is the result of MRR. There is a turning point at Top 20 and it reaches the optimal value; it remains stable until turning and falling at Top 50, and then it stabilizes.

5.3 Result Analysis of Different Relationship Types

To fully evaluate our algorithm, we analyzed the performance of our algorithm with different types of relationships (Table 3 & Figure 4). The statistical results show that the binding relationship type gets the best performance, followed by the interaction relationship type, and the effect relationship type gets the worst performance (Table 3). It shows that the binding relationship type has the highest performance on the head entity, while the interaction relationship type has a relatively balanced performance in the prediction of the head entity and the tail entity, and the effect relationship type has the lowest performance (Figure 4). The number of triples that are used to train maybe has an effect on the prediction performance. For example, the effect type with the least data obtains the lowest performance. However, the binding type has much less triples than the interaction type, and the head entity prediction effect of the binding type is still better than the interaction type. We analyzed the two types of head and tail entities and found that the

head entity of the binding relationship type is a virus protein, and the tail entity is a host protein. However, there are two types in interaction relationships. Namely, when the head entity belongs to a virus protein, the tail entity belongs to a host protein; when the head entity belongs to a host protein, the tail entity belongs to a virus protein. This may increase the prediction difficultness for the interaction relationships.

Table 3. Comparison results of link prediction methods.

Relation type	Train	Test	Head entity type	Tail entity type	MRR
Effect	24	3	Drug	Virus	0.007
Interaction	27860	3096	Protein	Protein	0.188
Produce	600	66	Virus	Protein	0.103
Binding	7516	835	Protein	Protein	0.205

In addition, to explore the reason of different prediction performances with different triples, we built a heterogeneous network containing all triples and calculated the edge betweenness centrality (an important edge attribute in complex network) of each triple. Therefore, we compared the ranking of test triples in predictive results with their betweenness in the network (Figure 5). In addition, for the prediction set, we divided the result ranking of the prediction head entity, the result ranking of the prediction tail entity, and the average of the two results, respectively, and calculated the Pearson correlation coefficient (PCC). The results showed that the rankings of triple are positively correlated to the edge betweenness (PCCs of the three are {0.406, 0.353, 0.446}, and the *P*-values are {2.272e-158, 1.158e-117, 2.256e-194}), i.e., when the triples are in the center of the network, it is difficult to rank them in the front position. On the contrary, triples at the edge of the network are easy to be ranked in the front position.

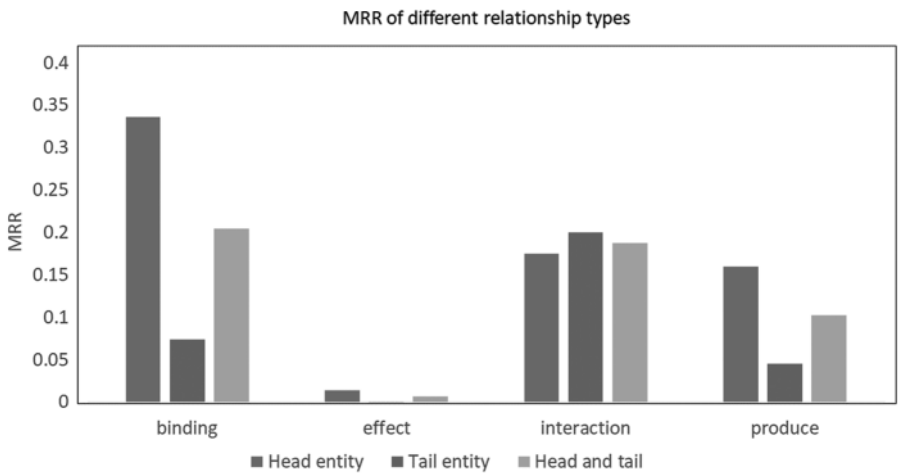

Figure 4. Comparison of prediction performance of different relationship types. For the prediction of head entities, the *binding* relationship type achieved the best performance, although the number of the edges of this type is not the largest. For the prediction of tail entities, the *interaction* relationship type with the largest amount of data obtains the best performance. But overall, the results of *interaction* are relatively stable. The *effect* relationship type is the worst because of the very small amount of data.

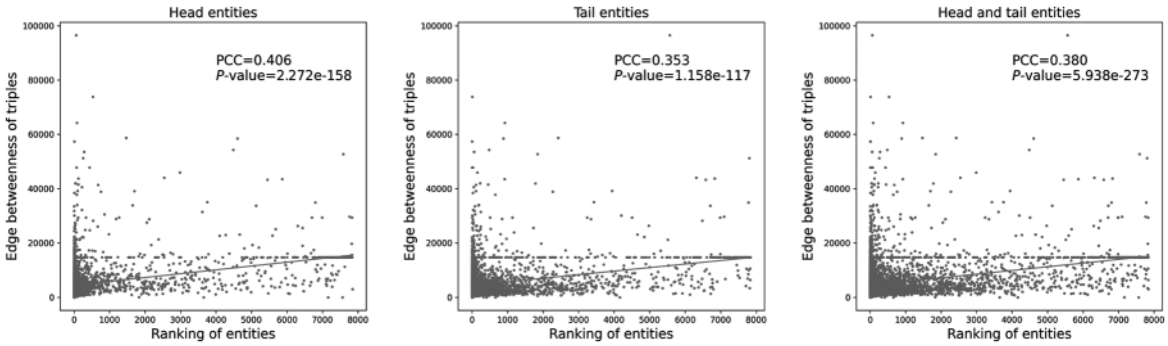

Figure 5. Correlation analysis between prediction triples and edge betweenness. From left to right are the ranking results of the prediction head entity, the ranking result of the prediction tail entity, and the average result of the two rankings of 4,000 data records in the prediction set. The abscissa is the ranking, the ordinate is edge betweenness, the red line in the figure is the fitting curve of the blue scatter distribution, and each figure is marked with Pearson result and *P*-value.

5.4 Case Study

To illustrate the predictive performance of our model, we took the two triples (?, interaction, ENTITY_6303) and (ENTITY_3946, binding, ?) in the first batch of prediction data as an example and obtained the top 10 candidate entities (Table 4). Bold entities are the hitting entities in benchmarks. $Rank_1$ represents the top 10 predicted entities of ComplEx-N3. $Rank_2$ represents the top 10 predicted entities of CP-N3. $Rank_3$ represents the top 10 predicted entities of our fusion strategy. In the top 10 candidate entities for (?, interaction, ENTITY_6303), we found that there are five entities, namely ENTITY_7358, ENTITY_5261, ENTITY_6303, ENTITY_2000 and ENTITY_1379 are actually recorded in the benchmarks. In the top 10 candidate entities for (ENTITY_3946, binding, ?), there are four entities (i.e., ENTITY_761, ENTITY_2386, ENTITY_6309 and ENTITY_3333) that are recorded in benchmarks. This demonstrated that the reliable integrated results are obtained by our model.

Table 4. Top 10 candidate entities for two predictive triples.

(?, interaction, ENTITY_6303)				(ENTITY_3946, binding, ?)			
Candidate entities	$Rank_1$	$Rank_2$	$Rank_3$	Candidate entities	$Rank_1$	$Rank_2$	$Rank_3$
ENTITY_7358	1	2	1	ENTITY_2675	1	1	1
ENTITY_5261	3	1	2	**ENTITY_761**	3	2	2
ENTITY_6303	2	4	3	ENTITY_6527	2	3	3
ENTITY_3101	5	3	4	ENTITY_3416	7	4	4
ENTITY_2000	4	5	5	ENTITY_1835	6	6	5
ENTITY_1379	6	7	6	**ENTITY_2386**	4	9	6
ENTITY_4654	8	6	7	**ENTITY_6309**	5	8	7
ENTITY_2459	7	12	8	**ENTITY_3333**	9	7	8
ENTITY_5750	9	10	9	ENTITY_425	12	5	9
ENTITY_7365	10	11	10	ENTITY_224	8	10	10

6. DISCUSSION

In this study, we proposed an effective fusion strategy to integrate the results of two tensor decomposition-based models, and the experimental results showed our model obtained high performance in LP task on KG. We added the list of entity attributes to our knowledge graph to get more attribute information of the entity and added reciprocal relations [13, 24] for every triple to expand the knowledge graph. In addition, there are still challenges in the LP task, including the need to design a knowledge graph embedding algorithm that integrates more ontology features, and analyze the equivalence between knowledge graph embedding and ontology reasoning. Future research might include: (i) combining logical rules which contain rich background information and KG triples in a unified KG completion framework [33, 34]; (ii) considering the path relationship between triples in the knowledge graph [35]. In the following work, we will continue to try new models and find the limitations of existing models to improve the performance. Of course, after analyzing the combination of prediction results and network attributes, we found that they are closely related, and we can combine network attributes for more exploration in the future.

ACKNOWLEDGEMENTS

This work was partially supported by Beijing Natural Science Foundation (No. M21012), National Key Research and Development Program of China (No. 2017YFC1703506, No. 2018AAA0100302) and National Natural Science Foundation of China (No. 82174533).

AUTHOR CONTRIBUTIONS

T. Jia (tingjia@bjtu.edu.cn) and Y.X. Yang (yuxia__yang@163.com) conducted the experiments and contributed to manuscript writing. X. Lu (lxrlj123@163.com) administered the data collection and supervised the data input process. Q. Zhu (qzhu@bjtu.edu.cn) contributed to manuscript writing. K. Yang (yangkuo@bjtu.edu.cn) coordinated the writing of the whole manuscript. X.Z. Zhou (xzzhou@bjtu.edu.cn) contributed to the final version of the manuscript. All authors have made meaningful and valuable contributions to revising and proofreading the manuscript.

REFERENCES

[1] Rossi, A., et al.: Knowledge graph embedding for link prediction: A comparative analysis. arXiv preprint arXiv:2002.00819 (2020)

[2] Wang, Q., et al.: Knowledge graph embedding: A survey of approaches and applications. IEEE Transactions on Knowledge & Data Engineering 29(12), 2724–2743 (2017)

[3] Lin, Y., et al.: Learning entity and relation embeddings for knowledge graph completion. In: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, pp. 2181–2187 (2015)

[4] Nathani, D., et al.: Learning attention-based embeddings for relation prediction in knowledge graphs. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 4710–4723 (2019)

[5] Bollacker, K., et al.: Freebase: A collaboratively created graph database for structuring human knowledge. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data (SIGMOD '08), pp. 1247–1250 (2008)

[6] Vrandecic, D., Krtoetzsch, M.: Wikidata: A free collaborative knowledgebase. Communications of the ACM 57(10), 78–85 (2014)

[7] Auer, S., et al.: DBpedia: A nucleus for a web of open data. In: International Semantic Web Conference & Asian Semantic Web Conference, pp. 722–735 (2007)

[8] Singha, A.: Official Google blog: Introducing the knowledge graph: Things not strings. Available at: https://www.blog.google/products/search/introducing-knowledge-graph-things-not/. Accessed 16 January 2022

[9] Jernigan, D.B.: Update: Public health response to the coronavirus disease 2019 outbreak—United States, February 24, 2020. Morbidity & Mortality Weekly Report 69(8), 216–219 (2020)

[10] Quinlan, J.: C4.5: Programms for machine learning. Morgan Kaufmann Publishers, San Francisco (1995)

[11] Asahara, M., Matsumoto, Y.: Japanese named entity extraction with redundant morphological analysis. In: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, pp. 8–15 (2003)

[12] Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back propagating errors. Nature 323, 533–536 (1986)

[13] Lacroix, T., Usunier, N., Obozinski, G.: Canonical tensor decomposition for knowledge base completion. In: Proceedings of the 35th International Conference on Machine Learning, pp. 2863–2872 (2018)

[14] Balažević, I., Allen, C., Hospedales, T.M.: TuckER: Tensor factorization for knowledge graph completion. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 5185–5194 (2019)

[15] Yang, B., et al.: Embedding entities and relations for learning and inference in knowledge bases. arXiv preprint arXiv:1412.6575 (2015)

[16] Trouillon, T., et al.: Complex embeddings for simple link prediction. arXiv preprint arXiv:1606.06357 (2016)

[17] Hitchcock, F.L.: The expression of a tensor or a polyadic as a sum of products. Journal of Mathematical Physics 6(1), 164–189 (1927)

[18] Zhang, C., et al.: Theoretical knowledge graph reasoning via ending anchored rules. arXiv preprint arXiv:2011.06174 (2020)

[19] Bordes, A., et al.: Translating embeddings for modeling multi-relational data. In: Proceedings of the 26th International Conference on Neural Information Processing Systems-Volume 2 (NIPS'13), pp. 2787–2795 (2013)

[20] Sun, Z., et al.: Rotate: Knowledge graph embedding by relational rotation in complex space. arXiv preprint arXiv:1902.10197 (2019)

[21] Qu, M., et al.: Rnnlogic: Learning logic rules for reasoning on knowledge graphs. arXiv preprint arXiv:2010.04029 (2020)

[22] Das, R., et al.: Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. arXiv preprint arXiv:1711.05851 (2017)

[23] Nguyen, D.Q., et al.: STransE: A novel embedding model of entities and relationships in knowledge bases. In: Proceedings of NAACL-HLT 2016, pp. 460–466 (2016)

[24] Dettmers, T., et al.: Convolutional 2d knowledge graph embeddings. arXiv preprint arXiv: 1707.01476 (2017)

[25] Schlichtkrull, M., et al.: Modeling relational data with graph convolutional networks. In: European Semantic Web Conference (ESWC), pp. 593–607 (2018)

[26] Guo, L., Sun, Z., Hu, W.: Learning to exploit long-term relational dependencies in knowledge graphs. In: International Conference on Machine Learning (ICML), pp. 2505–2514 (2019)

[27] Nickel, M., Tresp, V., Kriegel, H.P.: A three-way model for collective learning on multi-relational data. In: International Conference on Machine Learning, pp. 809–816 (2011)
[28] Avidan, S.: Ensemble tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence 29(2), 261–271 (2007)
[29] Araújo, M.B., New, M.: Ensemble forecasting of species distributions. Trends in Ecology & Evolution 22(1), 42–47 (2007)
[30] Zhou, X., Liu, J., Xiong, J.: Predicting distant metastasis in breast cancer using ensemble classifier based on context-specific mirna regulation modules. In: 2012 IEEE International Conference on Bioinformatics and Biomedicine, pp. 1–6 (2012)
[31] Balazevic, L., Allen, C., Hospedales, T.M.: Hypernetwork knowledge graph embeddings. arXiv preprint arXiv: 1808.07018 (2018)
[32] Zhang, W., et al.: Interaction embeddings for prediction and explanation in knowledge graphs. In: Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining (WSDM '19), pp. 96–104 (2019)
[33] Liu, H., Wu, Y., Yang, Y.: Analogical inference for multi-relational embeddings. In: Proceedings of the 34th International Conference on Machine Learning (ICML'17), pp. 2168–2178 (2017)
[34] Guo, S., et al.: Jointly embedding knowledge graphs and logical rules. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016), pp. 192–202 (2016)
[35] Qu, M., et al.: Rnnlogic: Learning logic rules for reasoning on knowledge graphs. arXiv preprint arXiv: 2010.04029 (2020)

AUTHOR BIOGRAPHY

Ting Jia received her undergraduate degree in Computer Science from Beijing Jiaotong University in 2019. She is currently a graduate student at the School of Computer and Information Technology, Beijing Jiaotong University. Her research interest is knowledge graph reasoning and medical knowledge engineering.

ORCID: 0000-0003-2830-5626

Yuxia Yang received her Master's degree in Computer Technology from Beijing Jiaotong University in 2021. During school, her research interest was learning to rank, and her main research directions were drug classification and drug redirection. She is currently an engineer at the Railway Group Information Technology Institute.

ORCID: 0000-0003-2035-9902

Xi Lu received his Master's degree in Computer Technology from Beijing Jiaotong University in 2021. During school, his research interest was data mining, and his main research direction was medical natural language text processing. He has conducted several experiments in medical text processing and co-published papers. He is currently a software development engineer in JD.

ORCID: 0000-0002-0387-5924

Qiang Zhu received her Ph.D. degree in Biomedical Engineering from Tianjin University in 2002. She is currently an Associate Professor with the School of Computer and Information Technology at Beijing Jiaotong University. Her research interests cover medical artificial intelligent, medical information processing, big data and data mining.
ORCID: 0000-0002-4677-1244

Kuo Yang received his Ph.D. degree in Computer Science from Beijing Jiaotong University in 2020. He is currently a lecturer with the School of Computer and Information Technology at Beijing Jiaotong University. His research interests cover medical data mining, network medicine, bioinformatics and artificial intelligence in medicine. He has published over 20 peer reviewed papers in journls including *Nucleic Acids Research* (NAR), *Journal of the American Medical Informatics Association* (JAMIA) and *IEEE/ACM Transactions on Computational Biology and Bioinformatics* (IEEE TCBB).
ORCID: 0000-0003-0736-4512

Xuezhong Zhou received his Ph.D. degree in Computer Science from Zhejiang University in 2005. He is currently a Professor with the School of Computer and Information Technology at Beijing Jiaotong University. His research interests cover medical data mining, network medicine, clinical decision support and medical knowledge engineering. He has published over 100 peer reviewed papers in journals including *Nature Communications*, *Ebiomedicine* and *Nucleic Acid Research* (NCR).
ORCID: 0000-0002-4713-3594